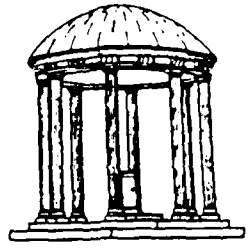
(2)

Two Adaptive Techniques Let Progressive Refinement Outperform the Traditional Radiosity Algorithm

TR89-020

August, 1989

John Airey
Ming Ouh-young



The University of North Carolina at Chapel Hill
Department of Computer Science
CB#3175, Sitterson Hall
Chapel Hill, NC 27599-3175

Two Adaptive Techniques Let Progressive Refinement Outperform the Traditional Radiosity Algorithm

John Airey Ming Ouh-young
Computer Science Department
University of North Carolina at Chapel Hill
January 1989

Abstract

Two adaptive techniques applied to form factor calculation in the progressive refinement version of radiosity allow it to compute the final converged result quicker than the traditional, matrix-solution-based, radiosity algorithm. The adaptive components of our algorithm are

1. adaptive subdivision of the hemi-cube/sphere as a function of delta form factor distribution,

2. adaptive reduction of hemi-cube/sphere resolution as a proportion of unshot radiosity in progressive refinement iterations and,

3. switching from a hemi-cube Z-buffer to ray sampling at the appropriate point in the computation to optimize efficiency.

The key idea is that of importance sampling. The reasoning is similar to the reasoning applied by Kajiya [Kajiya86] to eliminate unnecessary samples in traditional ray tracing and obtain the path tracing algorithm. The subdivision pattern and resolution of the hemi-cube are determined adaptively to keep the energy carried by each hemi-cube sample constant. This has the added benefit of reducing the error variance of the hemi-cube samples.

Our experimental evidence suggests that the progressive refinement approach must calculate twice the number of form factors as the traditional method before the process has converged. If we assume that form factor calculations consume greater than 90% of the computation, this means the progressive refinement approach takes twice as long to calculate the fully converged solution. However, the use of the above adaptive techniques can yield a 7-fold speedup of the progressive refinement approach, with the same error restrictions on the image, on data sets ranging from a couple hundred patches to thousands of patches. Thus the adaptive progressive refinement approach can compute a fully converged image 3-4 times quicker than the traditional radiosity algorithm.

1. Introduction

The progressive refinement radiosity algorithm or *shooting* algorithm introduced by Cohen et al. [Cohen88] is a major improvement over the traditional matrix solution based algorithm or *gathering* algorithm [Cohen85]. The quadratic space requirements of the traditional gathering algorithm made it impractical for large (real) data sets. The shooting algorithm has only linear space requirements. The addition of the progressive refinement property produces an algorithm suitable for use in a real-time design cycle. The only remaining disadvantage of the shooting algorithm is the increased time required to compute a converged solution. This disadvantage has been removed with the techniques described in this paper.

The central idea is to keep the amount of radiosity shot into the environment from a hemi-cube sample constant throughout the computation. A benefit of this is that the error introduced by each hemi-cube sample is likely to be similar; the sampling error variance is reduced. The nonadaptive shooting algorithm spends a lot of time on hemi-cube samples which contribute very little to the final result and, as a consequence, the sampling error variance may be large. This is analogous to the reasoning applied by Kajiya to produce the path-tracing algorithm. Kajiya noted that a lot of effort was spent on higher generation rays which contributed little to the image.

Consider the light shot from an emitter. The hemi-cube samples and thus the energy will be distributed over many patches. Each receiving patch will get only a fraction of the energy of the emitter and will contribute only that fraction of energy to the final image. However, each of the receiving patches will use a hemi-cube with resolution equal to the initial emitter. Thus it is possible for the contribution to the image per sample for each receiving patch to decrease exponentially. See figure 1 below.

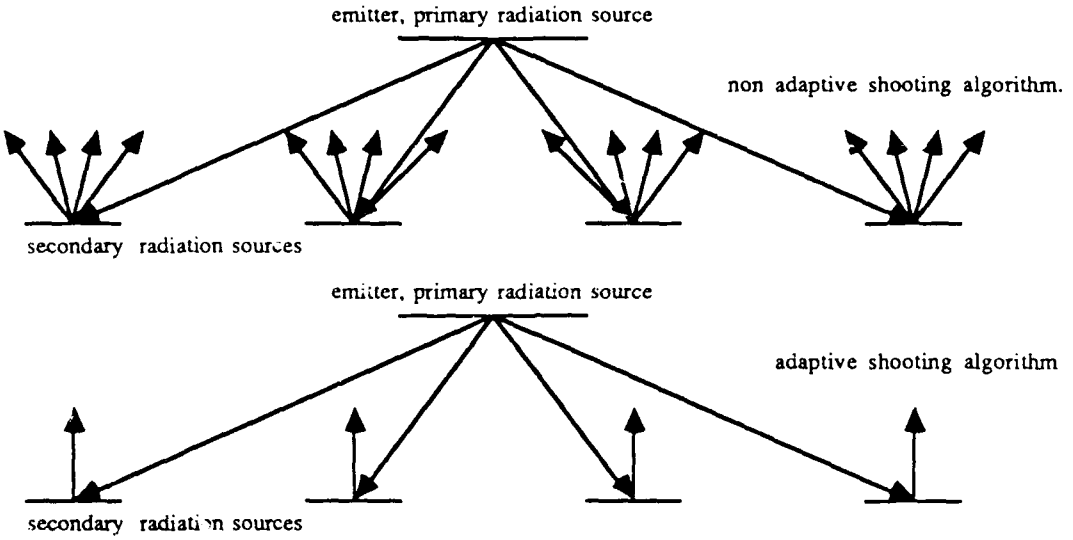


Figure 1. The non-adaptive shooting algorithm uses a constant sized hemi-cube. The radiation from the emitter, the primary radiation source, is spread over several secondary radiation sources. These secondary sources contribute only a fraction as much to the image as the primary radiation source but they use a hemi-cube with equal resolution. Thus, the contribution to the final image by the hemi-cube samples for each secondary radiation source will be greatly decreased. In contrast, the adaptive shooting algorithm adjusts the hemi-cube resolution as a function of the amount of energy it can shoot.

To ensure that each hemi-cube sample makes an equal contribution to the image we apply adaptive techniques in two areas. The first is the distribution of delta form factors over the hemi-cube. Ideally, each hemi-cube sample should have equal form factor values and thus shoot an equal amount of the unshot radiosity of the patch into the environment. However, the regular grid subdivision of the hemi-cube contains many samples which contribute very little to the result. By using a nonuniform subdivision of the hemi-cube we can reduce the number of samples and keep the error constant.

The largest delta form factor for a standard NxN hemi-cube is $4/(\pi N^2)$. This follows from the formula for the delta form factors,

Delta form factor = $1/\pi(x^2 + y^2 + 1)^2$ * Delta area for the top face,
Delta form factor = $z/\pi(y^2 + z^2 + 1)^2$ * Delta area for a side face

where Delta area = $4/N^2$ and x and y are 0 on the center of the top face.

Now, since the Delta form factors sum to 1, the inverse of the largest form factor yields the number of delta form factors necessary if all delta form factors are to have value equal to the largest form factor. Thus we need $\pi N^2/4$ samples. Since the number of samples in an NxN hemicube is $3N^2$ we need only $(\pi N^2/4) / (3N^2) = \pi/12$ as many samples. We describe the details of the non uniform subdivision in Section 2.

The second adaptive technique follows from the observation that the maximum unshot radiosity in each iteration drops off at a rate that can be fitted to an exponentially decaying function, see Figure

2 below. In particular, we have observed that after only 10% of the interations, the maximum unshot radiosity has dropped to 25% of its initial value. This is also reflected in the convergence graph depicted in Figure 7 of the original progressive radiosity paper [Cohen88]. Thus we can reduce the resolution of the hemi-cube as a proportion of unshot radiosity to keep the amount of radiosity shot per sample constant .

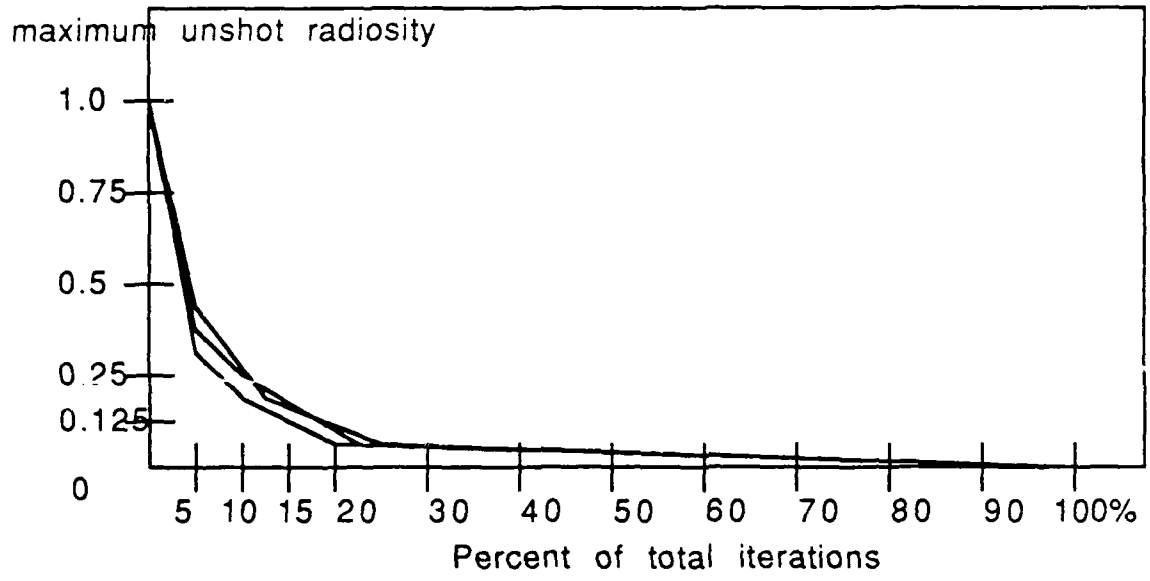


Figure 2. Typical curves of maximum unshot radiosity as a function of total iterations. This data is collected from eight data sets that range from 200 to 5000 patches. At 10% of the total iterations, the maximum unshot radiosity drops to approximately 1/4 of the initial maximum unshot radiosity. The average curve can be fit to an exponentially decaying function, $f(i) = E^i$, $0 <= i <= 100$ and $E = 0.87055$.

A consequence of the exponential drop off is that after a small fraction of the total iterations necessary for convergence, the hemi-cube resolution becomes quite small. The overhead of transforming and processing all the polygons in an environment for Z-buffer calculations dominates the scan conversion cost. This makes the use of sampling with rays more efficient when the hemi-cube resolution is small.

We have implemented both a Z-buffer based algorithm and a ray sampling algorithm which use the two adaptive strategies noted above. Error and speed analyses were performed to determine the speedup under given error restrictions. By comparing the two algorithms, we were able to deduce the efficiency of a hybrid algorithm which can outperform either algorithm used alone.

2. Implementation

We present the details of the adaptive sampling method used for the ray sampling method first because of its relative simplicity.

The hemi-cube is a construction which makes it possible to use a Z-buffer algorithm. It is derived from the Nusselt hemi-sphere analog [Cohen85]. This allows form factors to be computed as the fraction of the base of the hemi-sphere covered by the orthographic projection from the hemi-sphere. For this reason we can forget about the hemi-cube and concentrate on the hemi-sphere for purposes of ray sampling. Let X, the number of samples, be proportional to the unshot radiosity. We pick the constant of proportionality empirically as a runtime constant depending upon the desired accuracy of the image.

Next generate a subdivision of the the unit circle into X pieces. The form factor for each piece will be simply 1/X. We use polar coordinates to subdivide the unit circle. Let the number of theta divisions be some constant multiple (we use 2) of the number of radius divisions. The theta

divisions were spaced equally while the radius divisions were spaced as a function of the square root of the division number to keep areas equal among the divisions. The x,y coordinates of the center of the division were then used in the equation of the hemi-sphere, $z = \text{sqrt}(1 - x^2 - y^2)$, to get the z coordinate of the tip of the direction vector for the ray. Figure 3 depicts the nonuniform subdivision of the hemi-sphere and hemi-cube.

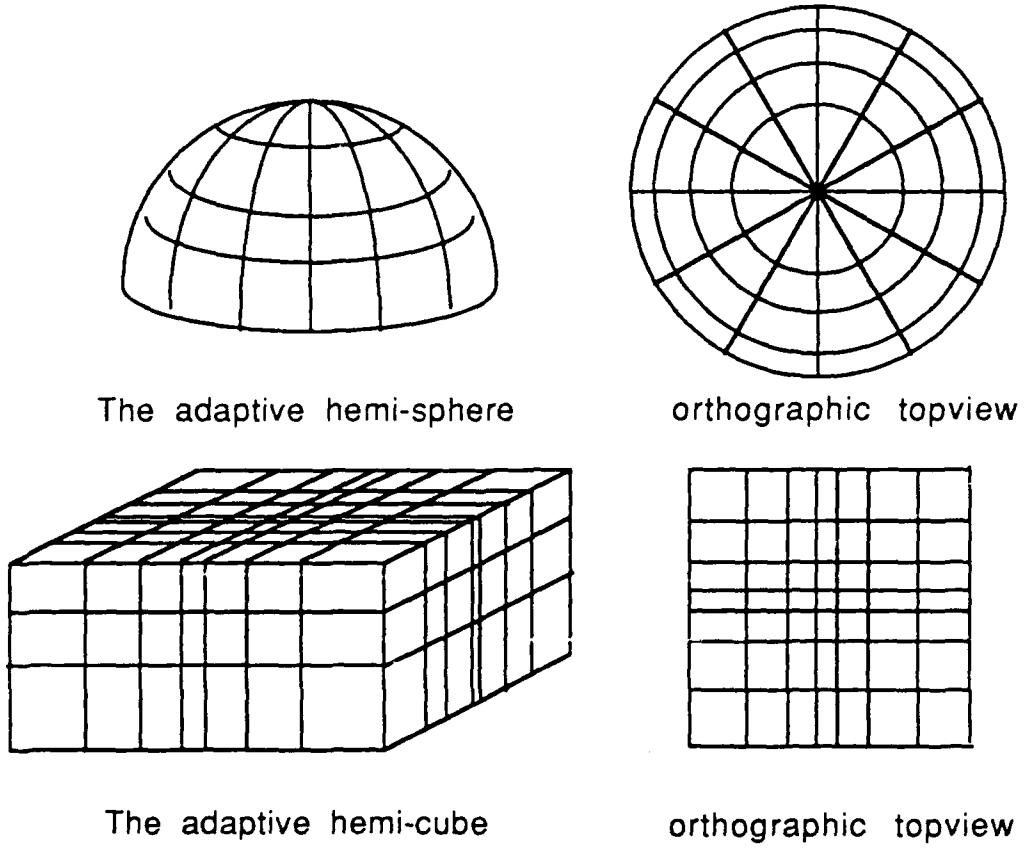


Figure 3. The nonuniform subdivision of the hemi-sphere and the hemi-cube as a function of the distribution of reflected light from a perfectly diffuse surface.

The ray intersection algorithm is optimized for environments where radiosity appears to be most useful, architectural models. The sets of polygons normal to each of the three coordinate axes are sorted by plane equation into three separate lists, one for each axis. This constitutes over 90% of the polygons in most of our models. A property of parallel planes and any ray is that the ray must intersect the planes in order along the normal to the parallel planes. A two dimensional analog is depicted in figure 4.

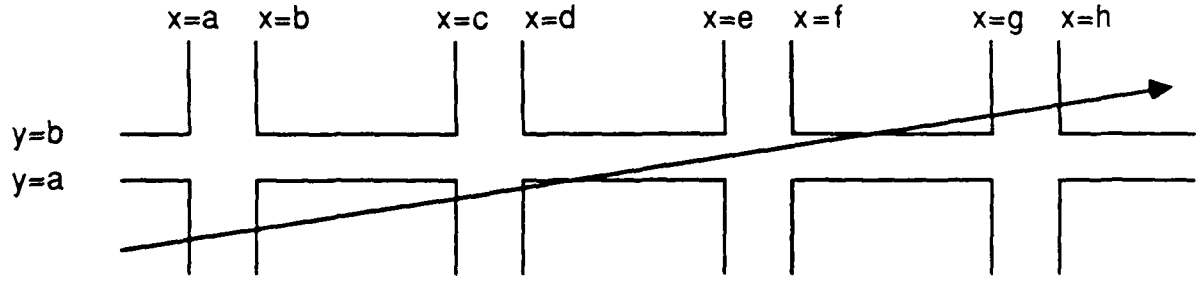


Figure 4. The diagonal ray intersects the horizontal line segments in vertical order through the parent lines y = a, then y = b. Similarly, the ray intersects the vertical line segments in horizontal order through the parent lines y = {a,b,c,d,e,f,g,h}. Merging the intersection order from each list yields the parameters of intersection along the ray in sorted order. This allows efficient ray intersection calculations.

Thus we have three seperate lists for which the intersection order is known independently. A three way merge allows us to step through the lists in order of intersection along the ray. The polygons which are not normal to one of the coordinate axes, *skew* polygons, are placed into a BSP tree [Naylor]. After an intersection is found with the axial polygons, the BSP tree is traversed from front to back until the previously found intersection point is passed or until a closer intersection point is found.

The Z-buffer based algorithm uses similar principles to adjust the resolution of the hemi-cube, although the construction of the nonuniform hemi-cube is more complicated than the construction of the nonuniform hemi-sphere. The number of samples is determined as a proportion of the unshot radiosity of the patch. N, the dimension of the top face, can be determined as follows,

N = sqrt(C /3 * unshot radiosity) where C is the desired radiosity per hemi-cube sample. N is rounded to the closest odd integer.

The width of the central pixel on the top face of the hemi-cube is thus $4/N^2$ and the x=y coordinate of the corner in the positive quadrant is 1/N. Next we generate nonuniform steps along the diagonal, y=x, of the front face of the hemi-cube, which determines the grid spacing.

From the equation for the Delta form factors, we may derive an iterative relation which can be used to generate the correct grid spacings, where x_i is the x and y coordinate along the diagonal of the grid,

$$(x_{i+1} - x_i)^2 / (2*((x_i + x_{i+1})/2)^2 + 1)^2 = 4/N^2 \quad \text{for } x_0 = 1/N.$$

A similar recurrence relation may be developed for the grid spacings for the side faces. In our implementation, we used an approximate solution to the above relation to get the grid spacings. A second pass calculated the exact delta form factors to use for our approximation.

The Z-buffer algorithm is identical to the traditional Z-buffer algorithm, except that a look-up table is generated which maps the regular grid to the nonuniform grid. When the boundaries of a span are determined using the regular grid spacing, the table is used to get the nonuniform grid spacings to use in the scan conversion.

3. Error Analysis

To compare the time performance of several variations of the radiosity algorithm it is desirable to establish that the image quality is comparable. However, it is difficult to compare two images quantitatively. It is known that the human visual system is much more sensitive to coherent error than incoherent error, or noise. An error metric that sums the absolute differences between patches in two images may not say anything about the differences the human visual system can discern. A statement such as, "image A is 10% different than image B", may mean many things depending upon how the error is distributed over the image.

In particular, the process of interpolating the patch values during display serves as a low pass filter which can remove the noise component. Our quantitative error analyses should only serve as an objective reference. The last word in image comparison should be human perception.

The test database for our quantitative error comparisons is the small "goral blocks" database with several cubes of varying colors. The database contained 232 patches. The refinement halted after 397 refinement iterations were computed, at which point the maximum unshot radiosity was less than 0.06 percent of the mean radiosity value. Nothing further can be shown on a 24 bit frame buffer. Two definitions of error were used in this analysis:

1) The average energy error is the summation of the energy difference between two patches divided by the total energy of the environment.

2) The total percent error is the summation of the radiosity differences between two patches divided by the radiosity of the patch. The average weighted percent error is the total percent error divided by the number of patches.

Three images best illustrate the results of our error analysis.

1) a 100x100 nonadaptive hemi-cube, which served as the reference image,

2) a 100x100 adaptive hemi-cube, and

3) a 100x100 nonadaptive jittered hemi-cube where each point sample was randomly positioned within the hemi-cube pixel rather than at the center of the hemi-cube pixel.

The results for these three images appear in Table 1 below.

	100x100 nonadaptive	100x100 adaptive	100x100 jittered
average energy error	0.0%	2.67%	4.4%
average weighted percent error	0.0%	4.92%	13.8%

Table 1. Error Analysis for a sample data set with 232 patches.

Although our error metrics indicate the images differ, we were unable to detect any difference in the images ourselves in the test images or in any other images. It appears that the effects of jittering are negligible. It increased the quantitative error but we were unable to detect the change in the image visually. We hypothesize that this is because the error is not coherent. It appears as noise which is filtered out by the process of interpolating patch values for display.

More important for the antialising of hemi-cube samples is the random rotation of the hemi-cube about its central axis recommended by Cohen et al. [Cohen85]. This is especially true of architectural models where most polygons are aligned to the coordinate axes. Hemi-cubes aligned with the coordinate axes can suffer heavy aliasing.

4. Speed Analysis

We have made several interesting empirical observations which allow us to construct a speed analyis of several radiosity solutions and suggest hybrid methods. These observations follow.

1) The preceding error analysis and our qualitative observations suggest that the image produced by the adaptive shooting algorithm is comparable to the the image produced by the non-adaptive shooting algorithm..

2) From collected data we observed that the unshot radiosity as a function of interation number closely matches an exponentially decaying function,

$$\text{decay(iteration)} = E^{\text{iteration percent complete}}, \text{ see Figure 2.}$$

After 10% of the total interations required for full convergence, the maximum unshot radiosity dropped to approximately 1/4th of the largest unshot radiosity, even after excluding light sources. For example, in the blocks data set used for the error analysis, after iteration 40 of 397, the adaptive approach was able to reduce the number of hemi-cube pixels to 1/4 of the starting number.

3) We have noted that the shooting algorithm must, on the average, compute twice as many form factors as the traditional gathering algorithm to compute a fully converged solution.

4) Although adapting the subdivision of the hemi-cube to the perfect diffuse reflection function reduces the number of necessary samples to $1/3.82 = \pi/12$, we have found that the overhead of transforming and scan converting the polygons makes it possible to get a speedup factor more than 2 only for a hemi-cube resolution greater than about 50x50. This limitation does not apply to the ray sampling method.

Given these assumptions, we can make a number of interesting conclusions about the performance of the adaptive shooting solution to the radiosity lighting model.

Figure 5 shows the cost of three approaches as a function of the percent of iterations completed. The non adaptive shooting approach is a horizontal line A, since each iteration takes essentially the same amount of time. The simple adaptive shooting aproach implemented with a Z-buffer is represented by line B. The cost is about 50% of A due to observation 4. If the resolution of the Z-buffer is decreased proportionally with the unshot radiosity, the result is curve C. Curve D depicts the performance of pure ray sampling which has true exponential decay since the cost is directly proportional to the number of ray samples.

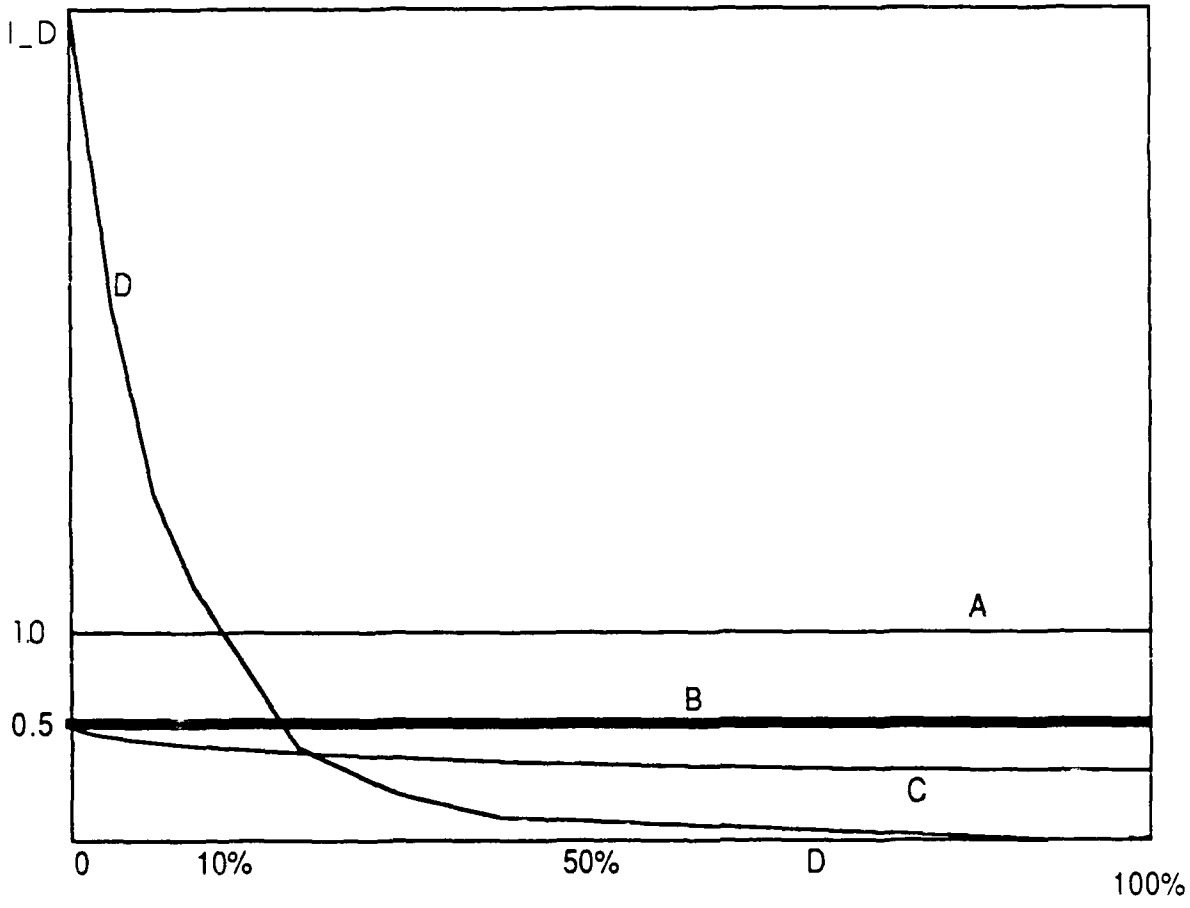


Figure 5. The cost of different methods as a function of percentage of total iterations.
Line A : The non-adaptive hemi-cube
Line B: The hemi-cube which is non-uniformly subdivided but which uses a constant resolution.
Line C: The fully adaptive hemi-cube.
Line D: fully adaptive ray sampling.
I_D is the relative cost of ray sampling vs the hemi-cube when an equivalent amount of rays are shot.

We may determine the cost of any method by taking the integral of the curve. If the curves intersect, the best method will be a hybrid method.
The optimal hybrid algorithm can be determined by using the method whose curve has least cost at any given iteration. A hybrid algorithm which starts with method C and switches to method D will give the optimal speedup. We analyze the costs of these methods and a few hybrid algorithms and compute the speedup next.

We begin by examing the cost of ray sampling, curve D, and the speedup over the nonadaptive

hemi-cube method, curve A. Let the decaying function of unshot radiosity be E^i, where i is the iteration number and E is determined by using observation 2. Thus E = 0.87055, since $0.87055^{10} = 1/4$.

Let the initial cost of the ray sampling method relative to the adaptive Z-buffer solution be I_D. Then the area under curve D is

$$\text{integral } (I_D * E^i) di \text{ from 0 to 100} = I_D / -\ln(E) = 7.21 * I_D$$

Thus the speedup of method D over method A is integral(A) / integral(D). Our empirical data shows I_D to range from 4 to 8, (observation 5) so the speedup ranges from 1.7 to 3.4.

We next examine the speedup gained from using a hybrid algorithm. After 15% of the total iterations are complete, the cost of ray sampling, curve D, will equal the cost of the nonadaptive hemi-cube, curve A, assuming the worst case observed value for I_D, 8. At this point we switch from hemi-cube sampling to ray sampling. The cost of of the ray sampling method over the last 85% of the iterations is $8*(E^{100} - E^{15})/\ln(E) = 7.21$. Thus the speedup gained in using method A then method D = (15 + 7.21) / 100 = 4.5.

If we replace method A by a faster method such as B, which is twice as fast due to observation 4, the speedup is (7.5 + 7.21) / 100 = 6.8. By replacing method B with method C we get a speedup of about seven. Although this analysis makes several assumptions based on empirical data, it demonstrates the importance of the decaying function of maximum unshot radiosity to efficiency.

5. Conclusion

We have presented improvements to the progressive refinement approach to radiosity that make it preferable to the traditional radiosity algorithm in all respects. Previously, the progressive refinement approach appeared to take twice as long as the traditional approach to generate a fully converged solution. We can now expect the adaptive shooting algorithm to take from 1/3 to 1/4 as long.

The traditional radiosity approach has been modified by several lines of research to incorporate non-diffuse components into the lighting model. The research by Wallace et al., [Wallace87], and M. Z. Shao et al., [Shao88] , are examples of research that is built on top of the diffuse solution. The latter effort heavily used the Gauss-Seidel matix solution to radiosity in their iterations. If the adaptive progressive refinement approach is the only way to deal with large numbers of patches, its efficiency is even more important to research that is based upon the radiosity solution for diffuse lighting. The very fact that the progressive radiosity approach is so much better than the traditional radiosity approach make all improvements to it that much more important.

While we expect that our results will have the greatest bearing upon software implementations run on uniprocessors which can handle adaptive strategies easily, these results do have ramifications for hardware implementations. We have concentrated upon keeping the image quality constant and reducing the time spent. However, another way to view the problem is to keep the time constant and improve the quality of the image. If the number of processors available for hemi-cube or hemi-sphere samples is necessarily a constant in a hardware SIMD system, this research shows why it is important to distribute the contribution to the image equally among samples to reduce the error variance and improve the image.

Acknowledgements

This research was aided greatly by many people at UNC. The other members of the Walkthru project and the Pixel-Planes project have been especially helpful.

Sample Images

Image 1a.

Images 1a and 1b. An office in Sitterson Hall at UNC CH. The lighting contributions for each of the two ceiling lights were computed independently by keeping a vector of unshot radiosities at each patch. Using the Pixel-Planes4 Prototype, we can adjust the weighting of the two lights seperately in real-time. Image 1a shows the front light at 90% intensity and the rear light at 10% intensity. Image 1b shows the front light at 20% and the rear light at 90%. This ability to adjust the lighting in real-time is useful to designers. This model contains roughly 2000 patches.

Image 2. Sitterson Hall hallway. Sitterson Hall, the UNC computer science dept. building has been modelled in AutoCad by members of the Walkthru project. The office above and this hallway are subsets of that database. The hallway has about 2000 patches.

Image 3. Church. This database has about 5000 2' patches and 20 light sources. It is based on an addition to an existing church in Chapel Hill which is under construction. The radiosity solution was calculated in less than 8 hours on a Sun-4. The Pixel-Planes4 Prototype is used to display the image which consists of about 10,000 triangles for an update rate of about 1/3 of a second.

References

[Cohen85]
Michael F. Cohen, Donald P. Greenberg
The hemi-cube: A radiosity solution for complex environments.
Siggraph 85, Proc., Vol. 19, No. 3, August 1985.

[Cohen88]
Michael F. Cohen, Shenchang Eric Chen, John R. Wallace, Donald P. Greenberg
A progressive Refinement Approach to Fast Radiosity Image Generation
Siggraph 88, Proc., Vol 22, No. 4, August 1988.

[Cook]
Robert L. Cook
Stochastic Sampling in Computer Graphics
ACM Transactions in Computer Graphics, Vol. 5, No. 1, January 1986.

[Kajiya86]
James T. Kajiya
The Rendering Equation
Siggraph 86, Proc., Vol. 20, No. 4, August 1986.

[Naylor]
Bruce F. Naylor
A Priori Based Techniques for Determining Visibility Priority for 3-D Scenes
Ph.D. Thesis, University of Texas at Dallas (May 1981).

[Wallace87]
John R. Wallace, Michael F. Cohen, Donald P. Greenberg.
A Two-Pass Solution to the Rendering Equation:
A Synthesis of Ray Tracing and Radiosity Methods
Siggraph 87, Proc., Vol. 21, No. 4, July 1987

[Shao88]
Min-Zhi Shao, Qun-Sheng Peng, You-Dong Liang
A New RAdiosity Approach by Procedural Refinements for Realistic Image Synthesis
Siggraph 88, Proc., Vol 21, No. 4, July 1987